On improving Recurrent Neural Network for Image Classification

B. Chandra*, Rajesh Kumar Sharma
Indian Institute of Technology Delhi,
New Delhi, India
bchandra@yahoo.co.in, justrks@gmail.com

***Abstract*—The paper proposes a new method for improving the performance of Recurrent Neural Networks. The proposed method uses two parallel recurrent layers which execute independent of each other. The final output of recurrent layer at any time step is computed as the mean of the modulus of the output of these two layers. The proposed method attempts to overcome the limitations of the existing Recurrent Neural Networks with regard to the flow of gradient. Comparative performance of the proposed method has been carried out with Long Short Term Memory (LSTM) and Identity initialized RNN (IRNN), the latest improved version of RNN for classification of images. On benchmark image datasets, it has been shown that the proposed method outperforms both IRNN and LSTM.**

Keywords— Recurrent Neural Network; Deep Learning; Activation Function

I. INTRODUCTION

Deep learning has shown significant advancement in several fields such as Computer Vision [1, 2], Speech Recognition [3, 4], Natural Language Processing [5, 6] and Biomedical Applications [7, 8]. Recurrent Neural Network (RNN) is one of most important subfield of Deep Learning which is used for handling sequential data. Due to sequential handling of data, RNN is inherently deep in time. RNN has achieved superior performance in several sequence learning tasks such as language translation [9] and natural language processing [10]. RNN is trained by backpropagation through time which suffers from the problem of vanishing or exploding gradient. In vanishing gradient problem, the norm of back propagated error gradient decreases exponentially with each time step, hence making it difficult to learn long term dependency in the input sequence. On the other hand, exploding gradient leads to large weight updates which makes training unstable.

Long Short Term Memory (LSTM) [11] has been proposed for solving the problem of exploding or vanishing gradient. LSTM removes the gradient problem by using various gates which control the flow of information in RNN. LSTM has outperformed standard RNN in numerous sequence learning tasks such as unsegmented connected handwriting recognition [12] and Automatic Speech Recognition [13]. There is a limitation of LSTM that the number of parameters is large compared to the standard RNN, which makes training of LSTM computationally expensive. Other Recurrent Neural Networks include Gated Recurrent Unit (GRU) [14] which was proposed to reduce the number of gates in LSTM. Recently Le et. al. [15] proposed a new RNN, termed as IRNN, which has been shown to perform at par with LSTM while using the same number of parameters as standard RNN. IRNN uses a standard RNN with Rectified linear activation function in the recurrent layer. The hidden to hidden recurrent weight matrix in IRNN is initialized as identity matrix, which enables the gradient to back propagate through long time steps without facing the problem of vanishing or exploding gradient.

IRNN has a limitation that the Rectified linear activation function truncates negative input at zero hence allowing the gradient to back propagate only when the corresponding input is positive. For the case where input is negative, information regarding the negative input is lost and the gradient does not flow through the rectified linear unit. This prevents the network from training efficiently.

A new Recurrent Neural Network has been proposed in the paper which removes the drawback of IRNN by using absolute value of the recurrent layer output. Absolute value provides non-linearity to the proposed RNN while preserving the flow of information during the forward propagation as well as during the backpropagation step. In addition, two recurrent layers are considered which execute independently of each other. The final activation of recurrent layer during any time step is taken as the arithmetic mean of these two independent layers after taking their modulus. The proposed RNN is termed as Parallel channel RNN (PC-RNN). An efficient weight initialization scheme has also been proposed for the PC-RNN. The performance of PC-RNN has been compared with that of IRNN on several benchmark image datasets, where each row of the image is considered as an input sequence. The classification results clearly show the superiority of PC-RNN over IRNN.

Authors thank the IIT Delhi HPC facility for computational resources. The Council of Scientific and Industrial research (CSIR) India, is thankfully acknowledged for providing research fellowship to Rajesh Kumar Sharma.

II. Recurrent Neural Networks in the Literature

Several types of Recurrent Neural Networks have been proposed in the literature viz. Long Short Term Memory (LSTM) [11], Gated Recurrent Unit (GRU) [14] and Identity based RNN (IRNN) [15]. The RNNs are described in details as follows.

A. Long Short Term Memory

A LSTM network uses various gates to control the input and output from the recurrent units. The equations for input gate i_t and forget gate f_t are given below where the corresponding weights are denoted by w and bias by b. Input is denoted by x.

$$i_t = s(w_x x_t + w_h h_{t-1} + w_c c_{t-1} + b_i)$$

$$f_t = s\left(w_{fx} x_t + w_{hf} h_{t-1} + w_{cf} c_{t-1} + b_f\right)$$

Where s denotes the sigmoid activation function.
The cell unit c_t is given by

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(w_{cx} x_t + w_{ch} h_{t-1} + b_c)$$

Where $\odot$ denotes term by term multiplication and tanh is used to limit $F(x_t, h_{t-1}) = w_{cx} x_t + w_{ch} h_{t-1} + b_c$ in the range (-1, 1). Output gate o_t is given by

$$o_t = s(w_{ox} x_t + w_{oh} h_{t-1} + w_{oc} c_t + b_o)$$

The hidden state of LSTM is given by.

$$h_t = o_t \odot \tanh(c_t)$$

Where tanh limits c_t in the range (-1, 1).

The output is computed as a linear combination of hidden states

$$y_t = w_{yh} h_t + b_y$$

The hidden to output weights are denoted by w_{yh}.

B. Identity Initialized RNN

Identity initialized RNN (IRNN) uses rectified linear activation function for hidden unit and initializes hidden to hidden recurrent matrix as identity. Given the input sequence x_t, the output sequence y_t is obtained as

$$h_t = relu(W_{xh} x_t + W_{hh} h_{t-1} + b_h)$$

$$y_t = W_{hy} + b_y$$

Where *relu* denotes Rectified Linear activation function. W and b denote corresponding weights and bias. IRNN uses identity to initialize W_{hh} and b_h is set as zero vector. Due to use of identity, each hidden state is directly obtained by copying previous hidden state and adding input term. Rectified linear activation sets all negative hidden states to zero. The identity initialization ensures that when the error derivative is back propagated through time, the error derivative remains constant assuming that input and bias are zero. Thus, the problem of vanishing gradient or exploding gradient does not occur in IRNN.

III. Proposed Method

The proposed method is termed as Parallel Channel RNN (PC-RNN) since it uses two parallel channels $h_t^{(1)}$ and $h_t^{(2)}$ given by.

$$h_t^{(1)} = x_t\, w_{xh}^{(1)} + h_{t-1} w_{hh}^{(1)} + b_h^{(1)} \quad (1)$$

$$h_t^{(2)} = x_t w_{xh}^{(2)} + h_{t-1} w_{hh}^{(2)} + b_h^{(2)} \quad (2)$$

The hidden layer activation is computed as follows.

$$h_t = \left(mod\left(h_t^{(1)}\right) + mod\left(h_t^{(2)}\right)\right)/2 \quad (3)$$

The output is given by

$$y_t = w_{hy} h_t + b_y \quad (4)$$

Here rather than initializing w_{hh} as identity as in IRNN, $w_{hh}^{(1)}$ and $w_{hh}^{(2)}$ are initialized such that the flow of gradient is maintained (see section 3.1). $b_h^{(1)}$ and $b_h^{(2)}$ are initialized as zero.

Training is performed using backpropagation with adaptive learning rate [16]. Though, the number of parameters for PC-RNN is more as compared to that of IRNN, the computations of the two channels in PC-RNN are independent of each other and can be implemented in parallel.

In IRNN, at time step t, h_t depends on h_{t-1} through identity matrix w_{hh} . If $input_t$ <0, the hidden layer activation $h_t = 0$, hence information regarding previous hidden state h_{t-1} is lost. In the proposed method, input information flows through both $h_t^{(1)}$ and $h_t^{(2)}$ and is never lost. In the proposed PC-RNN, the absolute value of the channels is used since a large negative value is more dominant than a small positive value if their absolute values are considered. The procedure for initialization of recurrent weight matrices is given in the following section.

A. Initialization of Recurrent Weight Matrices

The equation for recurrent layer activation is taken to be

$$h_t = \left(mod\left(h_{t-1}w_{hh}^{(2)}\right) + mod\left(h_{t-1}w_{hh}^{(1)}\right)\right)/2$$

assuming input and bias to be zero.

Let the activation of ith hidden neuron at time t be denoted by h_t^i. For simplicity, $w_{hh}^{(1)}$ and $w_{hh}^{(2)}$ are replaced by U and V.

$$h_t^i = \left(mod\left(\sum_j h_{t-1}^j U_{ji}\right) + mod\left(\sum_j h_{t-1}^j V_{ji}\right)\right)/2$$

The derivative of ith hidden neuron at time t with respect to kth hidden neuron at time t-1 is given as follows.

$$\frac{\partial h_t^i}{\partial h_{t-1}^k} = \frac{\left(\sum_j h_{t-1}^j U_{ji}\right)}{mod\left(\sum_j h_{t-1}^j U_{ji}\right)}\frac{U_{ki}}{2} + \frac{\left(\sum_j h_{t-1}^j V_{ji}\right)}{mod\left(\sum_j h_{t-1}^j V_{ji}\right)}\frac{V_{ki}}{2}$$

$$\frac{\partial h_t^i}{\partial h_{t-1}^k} = (\pm\, U_{ki} \pm\, V_{ki})/2 \quad (5)$$

When the gradient is +1 or -1, the vanishing or exploding gradient problem will not occur. Considering both +1 and -1, the following conditions are obtained for net back propagated gradient at kth neuron at time *t*-1.

$$mod\left(\sum_i \partial h_t^i / \partial h_{t-1}^k\right) = 1 \quad (6)$$

Using (5) and (6), we have,

$$mod\left(\sum_i(\pm\, U_{ki} \pm\, V_{ki})\right) = 2 \quad (7)$$

Gradient flow from the same recurrent neuron plays the most important role, hence taking both U and V as identity is a trivial solution of (7). However, using identity does not allow the gradient to flow from one recurrent neuron to a different recurrent neuron. In the proposed method, the matrices U and V are initialized randomly as diagonally dominant such that each entry in U and V is non-zero. The detailed procedure is given below.

$$U = P + a\, I_{J\text{x}J} \quad (8)$$

$$V = Q + a\, I_{J\text{x}J} \quad (9)$$

Where P and Q are matrices of order JxJ and each entry in P and Q follows $N(0,\sigma^2)$. J denotes the number of recurrent neurons, a is a scalar and I denotes identity matrix.

The following normalization procedure is used in order to make the matrices U and V as given in (8) and (9) to satisfy (7).

$$U_{ij} = 2\, U_{ij}/mod\left(\sum_{k=1:J} U_{ik} + V_{ik}\right)\ for\ i,j = 1\ to\ J \quad (10)$$

$$V_{ij} = 2\, V_{ij}/mod\left(\sum_{k=1:J} U_{ik} + V_{ik}\right)\ for\ i,j = 1\ to\ J \quad (11)$$

Initialization of U and V as Identity becomes a special case of (10) and (11).

Another variant of the proposed method is used in case of only single channel. The method is termed as SC-RNN. The weight U for the single channel is initialized as.

$$U = R + a\, I_{J\text{x}J}$$

$$U_{ij} = 2\, U_{ij}/mod\left(\sum_{k=1:J} U_{ik}\right) for\ i,j = 1\ to\ J \quad (12)$$

Where each entry of R follows $N(0,\sigma^2)$.

IV. RESULTS

Comparative performance evaluation has been carried out on several benchmark datasets MNIST [17], CIFAR-10 [18], CIFAR-100 [18], NORB [19] and SVHN [20]. The performance of PC-RNN and SC-RNN have been compared with LSTM and IRNN since LSTM is a widely known Recurrent Neural Network and IRNN is the latest development in Recurrent Neural Network. Description of datasets is given below in Table I.

TABLE I. DESCRIPTION OF DATASETS

Dataset	Image size	Training Samples	Validation Samples	Testing Samples	Classes
MNIST	28x28	50000	10000	10000	10
CIFAR10	3x32x32	40000	10000	10000	10
CIFAR100	3x32x32	40000	10000	10000	100
NORB	3x32x32	43740	4860	48600	5
SVHN	3x32x32	65931	7326	26032	10

A. Implementation Details

Results are obtained for all the variants of the proposed method viz. SC-RNN, PC-RNN. For the sake of comparison, the performance of PC-RNN has been evaluated both by initializing the recurrent weights as Identity (termed as PC-RNN1) and by initializing the recurrent weights as proposed in section III (termed as PC-RNN2). In PC-RNN1 and PC-RNN2, the parameter σ is kept fixed as 0.001 and a is fixed as 0.1 in (8), (9) and (10) for each of the datasets. In the case of LSTM and IRNN also, the randomly generated weights are initialized using normal distribution with mean 0 and standard deviation 0.001. Bias for all the methods is initialized as zero vector. The number of neurons in the recurrent layers is taken as 100. During training, if the gradient exceeds the range $[-1, 1]$, then the gradient is reset to the corresponding boundary value. Testing error corresponding to the epoch with least validation error is reported in Table II.

For classification of image datasets, i^{th} row of the image is considered as input for i^{th} time step of RNN. The output corresponding to the final row of the image is used to compute the cross entropy error, which is minimized by backpropagation. Training is performed for 500 epochs using batch size of 100. The classification error (%) for various datasets is given below.

TABLE II. CLASSIFICATION ERROR (%)

Dataset	LSTM	IRNN	PC-RNN1	PC-RNN2	SC-RNN
MNIST	1.61	2.0	1.5	1.47	1.7
CIFAR10	44.48	50.0	44.1	43.6	45.75
NORB	15.90	51.07	14.39	14.36	19.37
CIFAR100	72.71	81.52	75.4	72.96	75.54
SVHN	12.62	42.58	14.0	8.64	14.35

From Table II, it is seen that the performance of PC-RNN2 is much superior compared to IRNN. It also performs better than PC-RNN1. It can be seen that for both PC-RNN1 and PC-RNN2, there is drastic reduction in classification error in case of NORB and SVHN datasets. PC-RNN1 performs better than LSTM on majority of datasets and PC-RNN2 gives better performance than LSTM on all datasets except CIFAR100. Since SC-RNN uses only single channel, SC-RNN gives higher classification error compared to LSTM, PC-RNN1 and PR-RNN2 but the performance of SC-RNN is better than that of IRNN.

Based on the results, it can be asserted that using two parallel channels is superior to using only single channel. Also, from the results of PC-RNN1 and PC-RNN2, it can be seen that the proposed weight initialization performs better than using identity.

V. CONCLUSION

An improved RNN has been designed to reduce the vanishing gradient problem and to remove the problem of information loss in rectified linear activation function when input is negative. It is a promising method of using RNN for image classification.

REFERENCES

[1] Huang, G. B., Lee, H., & Learned-Miller, E. (2012, June). Learning hierarchical representations for face verification with convolutional deep belief networks. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on* (pp. 2518-2525). IEEE.

[2] Dong, C., Loy, C. C., He, K., & Tang, X. (2014). Learning a deep convolutional network for image super-resolution. In *Computer Vision–ECCV 2014* (pp. 184-199). Springer International Publishing.

[3] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... & Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *Signal Processing Magazine, IEEE*, *29*(6), 82-97.

[4] Deng, L., Li, J., Huang, J. T., Yao, K., Yu, D., Seide, F., ... & Gong, Y. (2013, May). Recent advances in deep learning for speech research at Microsoft. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on* (pp. 8604-8608). IEEE.

[5] Collobert, R., & Weston, J. (2008, July). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning* (pp. 160-167). ACM.

[6] Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013, October). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)* (Vol. 1631, p. 1642).

[7] Cruz-Roa, A. A., Ovalle, J. E. A., Madabhushi, A., & Osorio, F. A. G. (2013). A deep learning architecture for image representation, visual interpretability and automated basal-cell carcinoma cancer detection. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013* (pp. 403-410). Springer Berlin Heidelberg.

[8] Suk, H. I., & Shen, D. (2013). Deep learning-based feature representation for AD/MCI classification. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013* (pp. 583-590). Springer Berlin Heidelberg.

[9] Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., & Khudanpur, S. (2010, September). Recurrent neural network based language model. In *INTERSPEECH* (Vol. 2, p. 3).

[10] Mikolov, T., Kombrink, S., Burget, L., Černocký, J. H., & Khudanpur, S. (2011, May). Extensions of recurrent neural network language model. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on* (pp. 5528-5531). IEEE.

[11] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, *9*(8), 1735-1780.

[12] Graves, A., Fernández, S., Gomez, F., & Schmidhuber, J. (2006, June). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning* (pp. 369-376). ACM.

[13] Sak, H., Senior, A. W., & Beaufays, F. (2014, September). Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *INTERSPEECH* (pp. 338-342).

[14] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

[15] Le, Q. V., Jaitly, N., & Hinton, G. E. (2015). A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*.

[16] Zeiler, M. D. "ADADELTA: an adaptive learning rate method." *arXiv preprint arXiv:1212.5701* (2012).

[17] LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324.

[18] Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images.

[19] LeCun, Y., Huang, F. J., & Bottou, L. (2004, June). Learning methods for generic object recognition with invariance to pose and lighting. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on* (Vol. 2, pp. II-97). IEEE.

[20] Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., & Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning* (Vol. 2011, p. 4). Granada, Spain.